Institut
Teknologi
Sepuluh Nopember

TUGAS AKHIR – IS184853

RANCANG BANGUN MODUL PEMERINGKATAN KERENTANAN OTOMATIS DALAM SISTEM EGOVBENCH

DESIGN AND DEVELOPMENT OF AUTOMATIC RANKING MODULE FOR EGOVBENCH SYSTEM

Mochammad Rizki Wicaksono
NRP 05 2 1 13 4000 0072

Dosen Pembimbing
Bekti Cahyo Hidayanto, S.Si., M.Kom.
Nur Aini Rakhmawati, S.Kom,M.Sc.Eng,Ph.D.

DEPARTEMEN SISTEM INFORMASI
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2019

TUGAS AKHIR – IS184853

RANCANG BANGUN MODUL PEMERINGKATAN KERENTANAN OTOMATIS DALAM SISTEM EGOVBENCH

MOCHAMMAD RIZKI WICAKSONO
NRP 05 2 1 13 4000 0072

Dosen Pembimbing
Bekti Cahyo Hidayanto, S.Si., M.Kom.
Nur Aini Rakhmawati, S.Kom,M.Sc.Eng,Ph.D.

DEPARTEMEN SISTEM INFORMASI
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2019

FINAL PROJECT – IS184853

DESIGN AND DEVELOPMENT OF AUTOMATIC RANKING MODULE FOR EGOVBENCH SYSTEM

Mochammad Rizki Wicaksono
NRP 05 2 1 13 4000 0072

Supervisor
Bekti Cahyo Hidayanto, S.Si., M.Kom.
Nur Aini Rakhmawati, S.Kom,M.Sc.Eng,Ph.D.

INFORMATION SYSTEMS DEPARTMENT
Information Technology And Comunication Faculty
Sepuluh Nopember Institute of Technology
Surabaya 2019

LEMBAR PENGESAHAN

RANCANG BANGUN MODUL PEMERINGKATAN KERENTANAN OTOMATIS DALAM SISTEM EGOVBENCH

TUGAS AKHIR
Disusun untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Departemen Sistem Informasi
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember

Oleh :

MOCHAMMAD RIZKI WICAKSONO
NRP 05 2 1 13 4000 0072

Surabaya, Januari 2019

KETUA
DEPARTEMEN SISTEM INFORMASI

Mahendrawati ER., ST., M.Sc., Ph.D.
NIP 19761011 20060420 01

LEMBAR PERSETUJUAN

RANCANG BANGUN MODUL PEMERINGKATAN KERENTANAN OTOMATIS DALAM SISTEM EGOVBENCH

TUGAS AKHIR

Disusun untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Departemen Sistem Informasi
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember

Oleh :

MOCHAMMAD RIZKI WICAKSONO
NRP 05 2 1 13 4000 0072

Disetujui Tim Penguji : Tanggal Ujian : 12 November 2018
Periode Wisuda : Maret 2019

Bekti Cahyo Hidayanto, S.Si., M.Kom. **(Pembimbing I)**

Nur Aini Rakhmawati, S.Kom,M.Sc.Eng,Ph.D. **(Pembimbing II)**

Febrilliyan Samopa, S.Kom.,M.Kom,Dr.Eng **(Penguji I)**

Nisfu Asrul Sani, S.Kom, M.Sc **(Penguji II)**

DEPARTEMEN SISTEM INFORMASI

RANCANG BANGUN MODUL PEMERINGKATAN KERENTANAN OTOMATIS DALAM SISTEM EGOVBENCH

Nama Mahasiswa	**: Mochammad Rizki Wicaksono**
NRP	**: 05 2 1 13 4000 0072**
Jurusan	**: Sistem Informasi FTIK-ITS**
Pembimbing 1	**: Bekti Cahyo Hidayanto, S.Si., M.Kom.**
Pembimbing 2	**: Nur Aini Rakhmawati, S.Kom,M.Sc.Eng,Ph.D.**

ABSTRAK

Kerentanan web adalah salah satu kualitas yang sepatutnya dijaga oleh pengelola situs web. Banyak dari website pemerintah daerah tidak terurus dan menjadi sarang black-hat hacker yang membuat situs web tersebut menjadi terancam dalam penyebaran informasi yang dilakukan. Setidaknya ada 548 situs web pemerintah daerah yang ada di Indonesia, dan terdapat indikasi kerentanan web yang dikeola pemerintah daerah. Maka dari itu penulis akan membuat modul monitoring tentang kerentanan web yang akan di integrasikan dengan system EGov Bechmark yang sudah ada, mengunakan Web Vulnerability Scanner. Dari hasil yang didapatkan dari testing terhadap 12 pemda yang telah bekerja sama, dapat diketahui website pemda masih memiliki celah keamanan dalam kategori tinggi dan sangat banyak dalam ketegori medium. Temuan ini belum bisa menjadi tolak ukur keseluruhan namun TA ini setidaknya dapat melihat sebagian kualitas dari website pemerintah daerah saat ini.

Kata kunci: Web Security, Egov.

Halaman ini sengaja dikosongkan

DESIGN AND DEVELOPMENT OF AUTOMATIC RANKING MODULE FOR EGOVBENCH SYSTEM

Nama Mahasiswa	**: Mochammad Rizki Wicaksono**
NRP	**: 05 2 1 13 4000 0072**
Jurusan	**: Sistem Informasi FTIK-ITS**
Pembimbing 1	**: Bekti Cahyo Hidayanto, S.Si., M.Kom.**
Pembimbing 2	**: Nur Aini Rakhmawati, S.Kom,M.Sc.Eng,Ph.D.**

ABSTRACT

Web vulnerability is one of the qualities that should be maintained by website managers. Many of the local government websites are neglected that flourish black hat hackers and threatened the information integrity of local government website. There are at least 548 local government websites in Indonesia, and there are indications that the web is governed by the local government. Therefore the writer will make a monitoring module about studying the web which will be integrated with the existing EGov Bechmark system, using the Web Vulnerability Scanner. From the results obtained from the testing of 12 local geverment that have worked together, the website of the regional government can still have gaps in the high category and very much in the medium category. This finding cannot yet be a benchmark in its entirety, but this TA has been able to see part of the quality of the current local government website.

Keywords: ***Web Security, Egov.***

Halaman ini sengaja dikosongkan

KATA PENGANTAR

Alhamdulillah atas karunia, rahmat, barakah, dan jalan yang telah diberikan Allah SWT selama ini sehingga penulis mendapatkan kelancaran dalam menyelesaikan tugas akhir dengan judul:

RANCANG BANGUN MODUL PEMERINGKATAN KERENTANAN OTOMATIS DALAM SISTEM EGOVBENCH

Terimakasih atas pihak-pihak yang telah mendukung, memberikan saran, motivasi, semangat, dan bantuan baik materi maupun spiritual demi tercapainya tujuan pembuatan tugas akhir ini. Secara khusus penulis akan menyampaikan ucapan terima kasih yang sedalam-dalamnya kepada:

1. Bapak Dr. Ir. Aris Tjahyanto, M.Kom selaku Ketua Departemen Sistem Informasi ITS Surabaya.
2. Bapak Nisfu Asrul Sani, S.Kom, M.Sc selaku Ketua Prodi S1 Departemen Sistem Informasi ITS Surabaya dan selaku dosen penguji yang telah memberikan masukan untuk perbaikan tugas akhir ini.
3. Bapak Bekti Cahyo Hidayanto, S.Si., M.Kom. dan Ibu Nur Aini Rakhmawati, S.Kom,M.Sc.Eng,Ph.D. selaku dosen pembimbing yang meluangkan waktu, memberikan ilmu, petunjuk, dan motivasi untuk kelancaran tugas akhir ini.
4. Bapak Febrilliyan Samopa, S.Kom.,M.Kom,Dr.Eng selaku dosen penguji yang telah memberikan masukan untuk perbaikan tugas akhir ini.
5. Orang tua penulis, Jihad Santosa dan Astuti Orbaniatun yang telah mendokan dan mendukung dalam pengerjaan tugas akhir ini.
6. Seluruh dosen Jurusan Sistem Informasi ITS yang telah memberikan ilmu yang sangat berharga bagi penulis.
7. Rekan-rekan BELTRANIS yang telah berjuang bersama dalam menjalani perkuliahan di Jurusan Sistem Informasi ITS.
8. Rekan-rekan Admin Studio yang telah banyak membantu meringankan beban kepengurusan studio DSI.

9. Rekan-rekan yang sering berada pada Lab IKTI yang telah banyak membantu memberikan informasi tentang tugas akhir.
10. Berbagai pihak yang membantu dalam penyusunan Tugas Akhir ini dan belum dapat disebutkan satu per satu dengan dukungan, semangat, dan kebersamaan.

Penyusunan laporan ini masih jauh dari sempurna, untuk itu saya menerima adanya kritik dan saran yang membangun untuk perbaikan di masa mendatang. Semoga buku tugas akhir ini dapat memberikan manfaat pembaca.

Surabaya, November 2018
Penulis,

(Mochammad Rizki Wicaksono)

DAFTAR ISI

(Halaman ini sengaja dikosongkan)

DAFTAR GAMBAR

(Halaman ini sengaja dikosongkan)

DAFTAR TABEL

(Halaman ini sengaja dikosongkan)

DAFTAR KODE

(Halaman ini sengaja dikosongkan)

BAB I
PENDAHULUAN

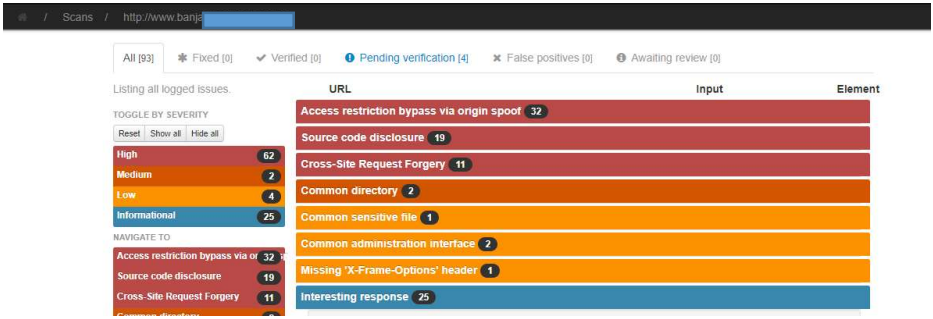
Pada bagian pendahuluan ini, akan dijelaskan mengenai latar belakang, masalah yang akan diselesaikan, batasan masalah, tujuan serta manfaat yang dihasilkan dari Tugas Akhir ini.

1.1 Latar belakang

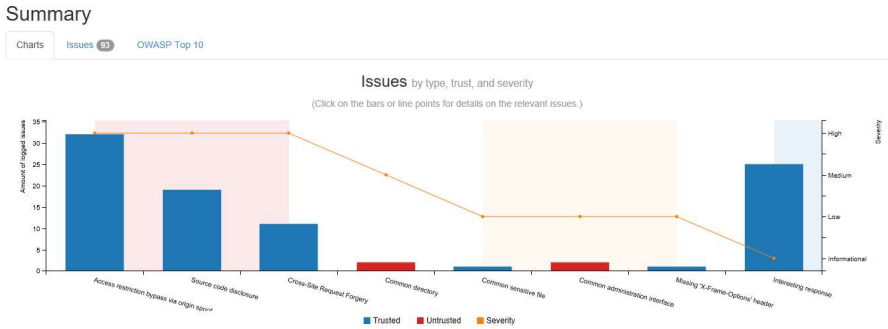
Pada beberapa tahun belakangan teknologi digital berkembang sangat pesat, berbagai kemudahan mengakses informasi menjadi bagian penting dari kehidupan sehari hari. Sehingga pemerintah berani menganggarkan biaya untuk pengelolaan website bagi pemerintah daerah, dengan harapan dapat menyampaikan informasi dengan cepat kepada masyarakat. Pada kenyataannya banyak website yang tadi sudah dianggarkan biayanya itu terbengkalai dan menjadi sarang *black-hat hacker* dalam melakukan uji kemampuan ataupun tujuan lain seperti digunakan untuk botnet.

EGov Benchmark adalah aplikasi untuk monitoring website pemerintah daerah (Provinsi, Kabupaten, dan Kota) yang ada di Indonesia. Dengan adanya aplikasi ini diharapkan dapat meningkatkan kesadaran pemerintah untuk meningkatkan mutu dan kualitas dari e-government milik pemerintah tersebut supaya lebih baik lagi dan masyarakat dan pemerintah ini dapat melihat perkembangan dari e-government pada masing-masing daerahnya. [1] Sebagai gambaran saat ini EGov Benchmark telah melakukan monitoring terhadap 548 situs web dari pemerintah daerah yang telah terdata.

Dari beberapa situs web yang ada dan telah terdata pada saat ini ternyata memiliki beberapa kelemahan yang dikategorikan *high*, seperti salah satu website dibawah ini yang memiliki celah *CSRF*, dan *Access Restriction Bypass*.



Gambar 1.1 WVS Report dari salah satu website pemerintah



Gambar 1.2 Chart Vulnerability pada salah satu web pemerintah

Namun, dari sistem EGov Benchamark yang sudah ada masih terdapat kekurangan yaitu masih belum ada monitoring bidang kerentanan, sehingga perlu untuk dikembangkan modul penilai kerentanan untuk monitoring website pemerintah daerah. Dengan adanya modul ini diharapkan pengelola website pemerintah daerah memerhatikan tingkat kerentanan dari website pemerintah sendiri, demi meningkatkan mutu dan kualitas dari rangkaian e-government milik pemerintah.

1.2 Rumusan masalah

Merujuk pada latar belakang yang telah dikemukakan sebelumnya, maka rumusan masalah pada tugas akhir ini adalah sebagai berikut:

1. Bagaimana tingkat keamanan dalam website pemerintah daerah?
2. Bagaimana pemeringkatan keamanan dalam website pemerintah daerah?

1.3 Batasan masalah

Dari permasalahan yang disebutkan di atas, batasan masalah dalam tugas akhir ini adalah:

1. Halaman yang dipindai adalah halaman yang berada pada domain utama.
2. Halaman yang dipindai hanya mencakup satu link dalam jangkauan crawler dari halaman utama.
3. Untuk alasan akurasi file file binary tidak di pindai.
4. Waktu yang digunakan untuk memindai suatu domain website adalah empat jam.

1.4 Tujuan

Berdasarkan hasil perumusan masalah dan batasan masalah yang telah disebutkan sebelumnya, maka tujuan dari tugas akhir ini adalah untuk mendapatkan informasi tentang kualitas keamanan dari website yang dioperasikan oleh pemerintah daerah.

1.5 Manfaat

Manfaat yang diharapkan dapat diperoleh dari tugas akhir ini adalah sebagai berikut:

1. Dapat membantu pemerintah daerah mengetahui kualitas keamanan website yang mereka kelola.
2. Dapat membantu pemerintah daerah mengetahui kelemahan website yang mereka kelola.
3. Dapat membantu pemerintah daerah dalam rekomendasi biaya untuk memperbaiki dana atau pemeliharaan website.

1.6 Relevansi

Relevansi tugas akhir ini terhadap laboratorium Infrastruktur dan Keamanan Teknologi Informasi karena tugas akhir ini berkaitan dengan beberapa mata kuliah yang berkaitan dengan laboratorium terkait yaitu: *Keamanan Aset Informasi*, *Teknologi Open Source dan Terbaru*, *Desain Manajemen Jaringan Komputer*, *Pemrograman Berbasis Web*.

BAB II
TINJAUAN PUSTAKA

Bab ini akan menjelaskan mengenai penelitian sebelumnya dan dasar teori yang dijadikan acuan atau landasan dalam pengerjaan tugas akhir ini. Landasan teori akan memberikan gambaran secara umum dari landasan penjabaran tugas akhir.

2.1 Penelitian Terdahulu

Berikut ini adalah daftar penelitian yang telah dilakukan sebelumnya yang mendasari penelitian tugas akhir ini :

1. Penelitian Mansour Alsaleh, Noura Alomar, Monirah Alshreef, dan Peneliti lain dari *King Abdulaziz City for Science and Technology* dan *King Saud University* mengenai "Performance-Based Comparative Assessment of Open Source Web Vulnerability Scanners": Pada penelitian ini penulis melakukan perbandingan dari *Web Vulnerabilities Scanner(WVS)* yang bersifat *Open Source* dalam memeriksa suatu celah keamanan dalam suatu website tertentu. Penelitian ini melihat bagaimana performa dari masing masing *WVS* dibandingkan dengan kelemahannya seperti waktu yang diperlukan untuk melakukan proses *scanning.*
2. Penelitian Jason Bau, Elie Bursztein, Divij Gupta, John Mitchell mengenai "State of the Art: Automated Black-Box Web Application Vulnerability Testing": Penelitian ini meneliti bagaimana kemampuan dari *Web Vulnerabilities Scanner(WVS)* pada tahun 2010 dalam menghadapi celah yang ada di website pada tahun tersebut dan berusaha melihat gap yang ada antara kemampuan dari WVS berbanding penyebaran celah yang ada, didapatkan hasil bahwa WVS yang ada cukup ampuh dalam mendeteksi *Cross-site Scripting(XSS)* dan *SQL Injection(SQLi)* dan *Information Disclosure*, Namun performanya jatuh

dalam XSS dan SQLi yang lebih '*advanced*', hasil dari tes *crawler* juga masih dirasa kurang pada saat itu. Penelitian ini juga meneliti penyebaran celah yang ada pada saat itu berdasarkan hasil pemindaian berbagai WVS tadi.

3. Penelitian Hsiu-Chuan Huang, Zhi-Kai Zhang, Hao-Wen Cheng, dan Shiuhpyng Winston Shieh dari National Chiao Tung University mengenai "Web Application Security: Threats, Countermeasures, and Pitfalls": Penelitian ini berisi cara mitigasi dari beberapa ancaman yang ada dengan implementasi keamanan yang baik namun hal tersebut ternyata belum cukup, karena banyak dari implementasi keamanan mengunakan 'signature' dan 'rule-based' yang membuat kemampuan pencegahannya masih dapat dimanipulasi sehingga celah keamanan masih dapat di akses.

Berikut adalah table perbandingan paper tentang Web Vulnerability Scanner:

Table 2.1 Paper WVS

Judul	**Tahun**	**Konten**	**Point Penting**	**Pengukuran**	**Domain**	**Target**
Performance-Based Comparative Assessment of Open Source Web Vulnerability Scanners	2017	Perbandingan WVS yang Open Source mencakup Arachni 0.4.3, Arachni 1.0.1, Wapiti 2.3.0, Skipfist 2.1	Memberikan gambaran tentang WVS open source yang terbaik. (dalam hal ini Arachni versi terbaru)	**Performance:** True positive rate(TPR), True negative rate(TNR), False positive rate(FPR), False negative rate(FNR),Positive predictive values(PPVs), Negative predictive values(NPVs), False omission rate(FOR), Accuracy, F-measure, Scanning speed, Crawler coverage, Vulnerability detection accuracy **Features:**	General	140 Distinct Web Based Real Applications

				Visualization features, Reporting features, Ease of configuration, Types of vulnerabilities that can be detected		
State of the Art: Automated Black-Box Web Application Vulnerability Testing	2010	Berisi berbagai macam WVS komersial dalam melakukan scanning, perbandingan performa dan pengunaan resource yang ada.	Melihat tipe serangan yang paling sering digunakan dalam serang yang dilakukan.	• Scanner Execution Time • Scanner Bytes Sent and Received • XSS Detection • SQL Injection Detection • Cross-Channel Scripting Detection • Session Management Vulnerability Detection • Cross-Site Request Forgery Detection • Information Disclosure Detection • Server and Cryptographic Configuration Vulnerability Detection • False Positive Count	General	Real web from VUPEN database

Using Web Security Scanners to Detect Vulnerabilities in Web Services	2009	Berisi perbandingan tiga produk komersial(empat dengan perbedaan versi) yang digunakan untuk web service testing, Mencakup	Bagaimana mendeteksi false positive dalam scanning di WVS.	• Vulnerabilities
				  ○ SQL injection
				  ○ XPath Injection
				  ○ Code Execution Buffer
				  ○ Overflow
					Tech provider (Microsoft, Google and Xara), General Bussiness.	300 REAL publicly availabl e services

		Accunetix, IBM Rational AppScan, HP WebInspect		○ Username/Password Disclosur ○ Server Path Disclosure • False Positive Percentage • WVS Coverage for SQL injection		

2.2 Dasar teori

Penjelasan bagaimana dasar teori yang digunakan dalam tugas akhir ini.

2.2.1 Web Vulnerabilities

Vulnerabilities dalam dunia komputer adalah sebuah kelemahan yang memberikan kemampuan bagi penyerang untuk mengurangi kemampuan suatu sistem dalam menyajikan, mengunakan dan memproses informasi yang ada atau dalam kata lain mengurangi kemampuan *Information Assurance* dari sistem. [2] [3]

Websites (Web) adalah tempat data dan informasi yang didefinisikan dengan *Uniform Resource Locators,* yang saling terhubung mengunakan *hyperlink* dan dapat diakses mengunakan *Internet*. [4]

Web Vulnerabilities adalah vulnerabilities yang terletak dalam teknologi web.

2.2.1.1 SQL Injection

SQL Injection adalah sebuah teknik yang menyalahgunakan sebuah celah keamanan yang terjadi dalam lapisan basis data sebuah aplikasi. Celah ini terjadi ketika masukan pengguna tidak disaring secara benar dari karakter-karakter *escape* dalam string yang digunakan dalam pernyataan SQL atau masukan pengguna tidak *Strongly typed* dan karenanya dijalankan tidak sesuai harapan. Ini sebenarnya adalah sebuah contoh dari sebuah kategori celah keamanan yang lebih

umum yang dapat terjadi setiap kali sebuah bahasa pemrograman. [5] [6]

2.2.1.2 Cross-site Scripting

Cross-site Scripting (XSS) adalah jenis injeksi, di mana skrip berbahaya disuntikkan ke situs web sehingga menyebabkan informasi menjadi tidak sesuai dan terpercaya. Serangan XSS terjadi saat penyerang menggunakan aplikasi web untuk mengirim kode berbahaya, umumnya berupa *browser side script*, ke pengguna akhir lainya. Serangan ini cukup meluas dan terjadi di mana saja aplikasi web menggunakan masukan dari pengguna dan menampilkan keluaran yang dihasilkannya tanpa memvalidasi atau *encoding*. [7]

2.2.1.3 Cross-site Request Forgery

Cross-Site Request Forgery (CSRF) adalah serangan yang memaksa pengguna akhir melakukan tindakan yang tidak diinginkan pada aplikasi web sebagai pengguna akhir. Serangan CSRF secara khusus menargetkan *state-changing requests*, bukan pencurian data, karena penyerang tidak memiliki cara untuk melihat respons terhadap permintaan palsu tersebut. Dengan sedikit bantuan *social engineering* (seperti mengirim tautan *via* email atau *chat*), penyerang dapat mengelabui pengguna aplikasi web untuk melakukan tindakan penyerang yang dipilih. Jika korban adalah pengguna normal, serangan CSRF yang berhasil dapat memaksa pengguna melakukan permintaan perubahan status seperti mentransfer dana, mengubah alamat email mereka, dan sebagainya. Jika korban adalah *administrator*, CSRF bisa mengkompromikan seluruh aplikasi web. [8]

2.2.1.4 File Inclusion

File inclusion adalah jenis kerentanan yang paling sering ditemukan untuk mempengaruhi aplikasi web yang mengandalkan *scripting runtime*. Ini menjadi masalah saat aplikasi ini membuat path kearah file yang masukan. Sehingga file yang masukan dapat di eksekusi oleh aplikasi saat *runtime.* Biasanya apabila celah ini berhasil

di eksploitasi maka akan menghasilkan *vulnerable* lain yaitu *remote code execution* [9]

1. Local File Inclusion

   Local File Inclusion (LFI) mirip dengan kerentanan Remote File Inclusion kecuali daripada memasukkan file *remote*, hanya file lokal yaitu file pada server saat ini yang dapat disertakan untuk eksekusi. Masalah ini masih dapat menyebabkan *remote code execution* dengan menyertakan file yang berisi data penting seperti log akses server web atau list password. [9]

2. Remote File Inclusion

   Remote File Inclusion (RFI) terjadi saat aplikasi web mendownload dan mengeksekusi file remote. File jarak jauh ini biasanya didapat dalam bentuk HTTP atau FTP URI sebagai parameter yang disediakan pengguna ke aplikasi web. [9]

2.2.1.5 Unvalidated Redirects and Forwards

Unvalidated Redirects and Forwards mungkin dilakukan saat aplikasi web menerima masukan yang tidak tepercaya yang dapat menyebabkan aplikasi web mengarahkan ulang permintaan ke URL yang berisi masukan yang tidak tepercaya. Dengan memodifikasi masukan URL yang tidak dipercaya ke situs berbahaya, penyerang dapat berhasil meluncurkan penipuan phishing dan mencuri kredensial pengguna. Karena nama server di link yang dimodifikasi identik dengan situs aslinya, upaya phishing mungkin memiliki tampilan yang lebih dapat dipercaya. Serangan juga dapat digunakan untuk membuat URL yang seharusnya tidak ada dan dapat melewati cek akses kontrol aplikasi dan kemudian meneruskan penyerang ke fungsi istimewa yang biasanya tidak dapat diakses. [10]

2.2.1.6 Unprotected Backup Files

Unprotected Backup Files adalah sekumpulan file yang biasanya digunakan dalam tahap development dari aplikasi web, file ini biasanya berisi *debug, inner schema*, *backdoors*, *administrative interface, source*

code, maupun *credentials* untuk *administrative interface*. [11] [12]

2.2.1.7 Path Traversal

Path Transversal adalah vulnerability yang bertujuan mencari akses file dan direktori yang disimpan diluar dari *web root folder,* dengan memanipulasi variable yang mereferensikan files seperti *dot-dot-slash* (../) dan variasinya atau mengunakan *full file path (absolute file path)* dengan begitu penyerang berhasil melakukan akses terhadap file yang disimpan didalam sistem operasi termasuk source code aplikasi web. [13]

2.2.1.8 Command Injection

Command injection adalah serangan dimana tujuannya adalah eksekusi perintah sewenang-wenang pada sistem operasi host melalui aplikasi yang rentan. Serangan injeksi perintah dimungkinkan saat aplikasi melewati data yang dimasukan pengguna yang tidak aman (formulir, cookies, header HTTP dan lain lain) ke *shell* dari sistem. Serangan *command injection* sebagian besar dimungkinkan karena validasi masukan yang tidak mencukupi. [14]

2.2.1.9 Code Injection

Code Injection adalah istilah umum untuk jenis serangan yang terdiri dari kode suntik yang kemudian ditafsirkan / dieksekusi oleh aplikasi. Jenis serangan ini memanfaatkan penanganan data yang tidak dipercaya. [15]

2.2.2 Web Vulnerabilities Scanner

Web Vulnerabilities Scanner (WVS) adalah alat otomatis untuk memindai aplikasi web yang biasanya digunakan untuk mencari celah keamanan seperti *Cross-site Scripting(XSS), SQL Injection(SQLi), Command Injection(CI), Path Tranversal, Local File Inclusion(LFI), Remote File Inclusion(RFI)* dan Konfigurasi server yang tidak aman.

Berikut adalah perbandingan dari beberapa WVS yang telah dikumpulkan dari beberapa sumber baik paper ataupun benchmark yang berhasil penulis simpulkan.

Table 2.2 WVS Benchmark

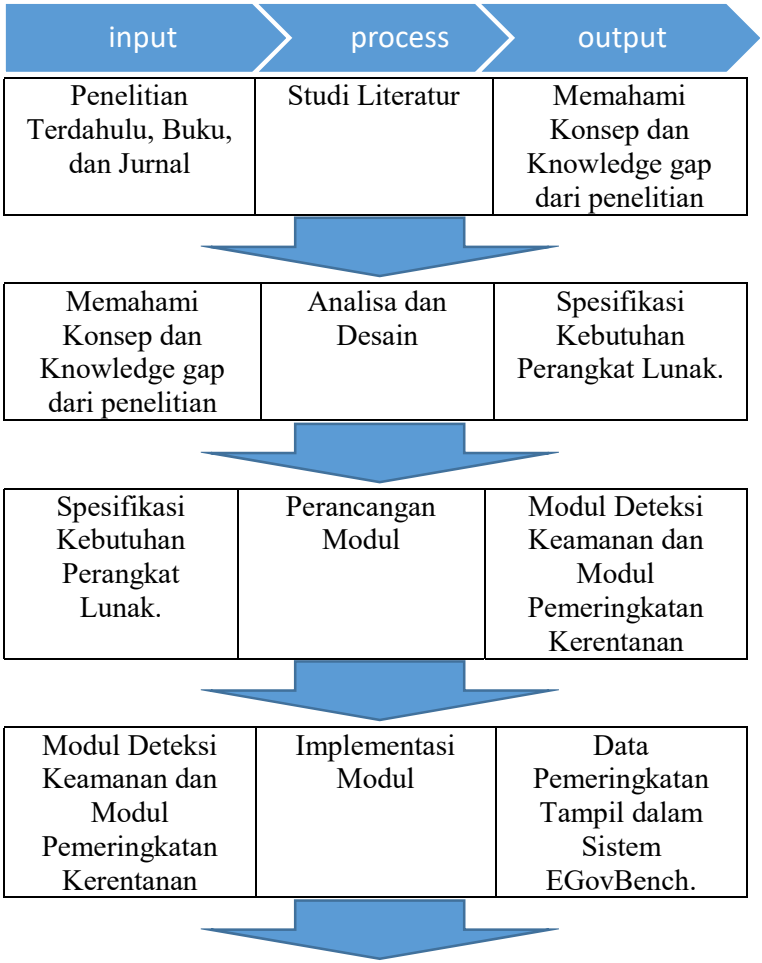
Nama	Lisensi	Platform	Komparasi*							Note Penulis
				WIVET	SQLi	RXSS	LFI	RFI	Redirect	Backup	
Acunetix WVS	• Commercial • Free (Limited Capability)	Windows	Accuracy	94.00 %	100.00%	100.00%	94.12 %	100.00%	100.00%	32.61 %	Stable
			False+		0.00 %	0.00 %	0.00 %	0.00 %	11.11 %	0.00 %	-
Arachni	• Free / Open Source Scanners	Windows Linux MacOS	Accuracy	96.00 %	100.00%	90.91 %	100.00%	100.00%	100.00%	100.00%	Stable, RPC / REST API
			False+		0.00 %	0.00 %	0.00 %	0.00 %	0.00 %	0.00 %	-
W3AF	• Free / Open Source Scanners	Linux Unix MacOS	Accuracy	19.00 %	35.29 %	37.88 %	57.48 %	16.67 %	63.33 %	22.83 %	-
			False+		30.00 %	0.00 %	12.50 %	16.67 %	11.11 %	0.00 %	Unstable
OWASP ZAP	• Free / Open Source Scanners	Windows Linux MacOS	Accuracy	73.00 %	100.00%	100.00%	75.00 %	100.00%	16.67 %	38.04 %	Stable, Very Good REST API
			False+		30.00 %	0.00 %	0.00 %	16.67 %	0.00 %	33.33 %	Huge Memory Usage

*Diambil dari http://www.sectoolmarket.com/price-and-feature-comparison-of-web-application-scanners-unified-list.html

Dari table diatas maka dapat di ambil kesimpulan penulis akan mengunakan WVS Arachni dalam melakukan pemindaian website target.

BAB III
METODOLOGI PENGERJAAN TUGAS AKHIR

Pada bab metode penelitian akan dijelaskan mengenai tahapan – tahapan apa saja yang dilakukan dalam pengerjaan tugas akhir ini beserta deskripsi dan penjelasan tiap tahapan tersebut. Lalu disertakan jadwal pengerjaan tiap tahapanan.



Data Pemeringkatan Tampil dalam Sistem EGovBench.	Testing Aplikasi	Dokumentasi testing aplikasi.

Pada bab ini akan dijelaskan tentang metodologi yang akan digunakan dalam penyusunan tugas akhir. Metodologi akan digunakan sebagai panduan dalam penyusunan tugas akhir agar terarah dan sistematis.

3.1 Studi Literatur

Pada tahap ini dilakukan pengumpulan literatur yang mendukung dalam menyelesaikan tugas akhir ini. Literatur disini adalah penjelasan konsep – konsep atau penelitian sebelumnya yang pernah dilakukan dan didokumentasikan dalam buku, jurnal, maupun website. Output atau keluaran proses ini adalah pemahaman mengenai konsep dan knowledge gap pada penelitian sebelumnya.

3.2 Analisa dan Desain Modul

Pada tahap ini dilakukan analisa dan desain modul yang akan di buat, yaitu bagaimana memeriksa keamanan website pemerintah yang akan dinilai dan bagaimana cara memberikan pembobotan nilai pada website pemerintah tersebut. Pada tahap analisa desain sistem ini yang harus dilakukan adalah melakukan analisa dan mendaftar kebutuhan fungsional apa saja yang akan di buat, kebutuhan funsional tersebut terbagi menjadi dua yaitu:

a) Bagian modul keamanan :
    i) Melakukan pemindaian keamanan kepada website pemerintah.

ii) Scheduler melakukan penjadwalan pemindaian website target, berdasarkan waktu dan *instance* pemindaian yang berjalan.

iii) Mengirimkan hasil pemindaian kedalam *server storage*.

iv) Mengirimkan data yang disimpan kedalam modul penilaian.

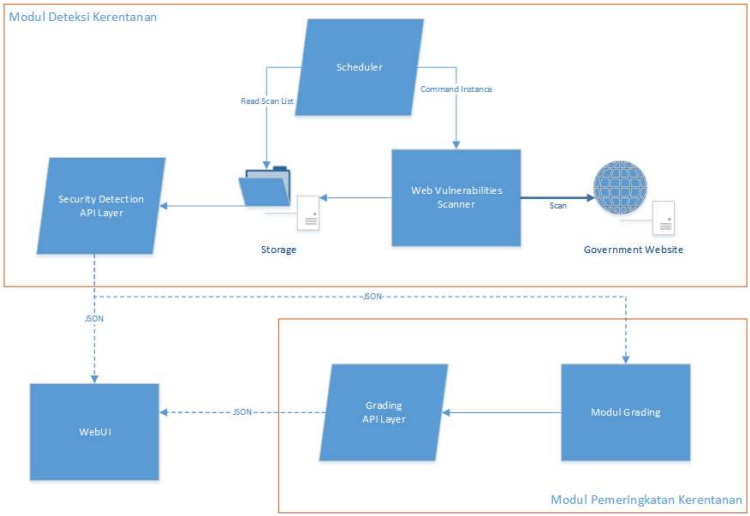
b) Bagian modul penilaian :

i) Menentukan nilai dengan mengacu pada jumlah deteksi kerentanan perkategori.

ii) Melakukan perhitungan nilai berdasarkan hasil yang didapatkan pada modul keamanan.

3.3 Perancangan Sistem

Pada tahap ini akan dimulai untuk melakukan perancangan modul sehingga server dapat melakukan pemindaian website pemerintah dan memberikan nilai terhadap keamanan dari website tersebut. Bagan dibawah merupakan rancangan arsitektur sistem yang akan dibangun.



Gambar 3.1 Desain Sistem

Pada Bagan diatas terlihat bahwa Web Vulnerabilities akan digunakan untuk melakukan pemindaian kepada website target dan akan mengirimkan hasil pemindaian kedalam storage server. Setelah selesai mengirimkan kedalam storage, data tersebut dapat dipanggil kedalam WebUI atau kedalam Modul *Ranking* menggunakan API yang akan mengirmkan data berbentuk JSON.

3.3.1 Modul Ranking

Modul *Ranking* akan melakukan penilaian berdasarkan data yang didapatkan dari Modul *Security Detection,* dengan perangkingan menggunakan ranking per kategori dan ranking overall berdasarkan nilai perkategori.

Pemeringkatan berdasarkan kategori ini dilihat dalam setiap kategori. Untuk kategori High seperti table dibawah ini merupakan contoh pemeringkatan pada kategori kerentanan HIGH dimana semakin banyak ditemukan kerentanan peringkatnya semakin rendah.

Table 3.1 Sebelum di Ranking

Target Website	High
Web A	**1**
Web B	**1**
Web C	**4**
Web D	**0**

Table 3.2 Sesudah di Ranking

Rank	Target Website	High
1	**Web D**	**0**
2	**Web B**	**1**
3	**Web A**	**1**
4	**Web C**	**4**

Pemeringkatan *Overall* adalah kombinasi dari pemeringkatan kategori. Dimana pemeringkatan ini dilakukan dengan cara memeringkatkan berdasarkan jumlah dari prioritas kategori, Prioritas kategori adalah dengan urutan berdasarkan tingkat kerawanan dari yang paling berbahaya yaitu: *HIGH, MEDIUM, LOW, INFORMATIONAL*. Semakin berbahaya suatu website maka semakin rendah posisi dipemeringkatan secara overall. Jadi Contoh untuk pemeringkatan *overall* situs web adalah sebagai berikut:

Table 3.3 Sebelum di Ranking

Kategori Deteksi	High	Medium	Low	Informational
Web A	1	4	100	17
Web B	1	2	50	20
Web C	4	2	7	10
Web D	0	0	1	21

Table 3.4 Sesudah di Ranking

Rank	Kategori Deteksi	High	Medium	Low	Informational
1	Web D	0	0	1	21
2	Web B	1	2	50	20
3	Web A	1	4	100	17
4	Web C	4	2	7	10

Kemudian ada fitur rekomendasi untuk perbaikan website, jika website yang di pindai memiliki kerentanan yang cukup tinggi misalnya memiliki kerentanan *HIGH* maka akan direkomendasikan untuk perbaikan celah yang ada dengan prioritas sesegera mungkin. Berikut adalah penggolongan rekomendasi web yang perlu diperbaiki:

Kategori	Rekomendasi	Rekomendasi Tambahan Biaya Untuk Perbaikan*
Sangat Rawan	Segera perbaiki situs web yang ada karena memiliki kerentanan yang sangat tinggi	RP 13.532.000.0
Rawan	Direkomendasikan ada perbaikan situs web yang ada dalam tahun	RP 6.766.00.0

	ini karena memiliki kerentanan yang cukup tinggi	
Sedikit Rawan	Direkomendasikan untuk memantau secara berkala keadaan kerentanan yang ada.	Sesuai Kebijakan Pemda Terkait

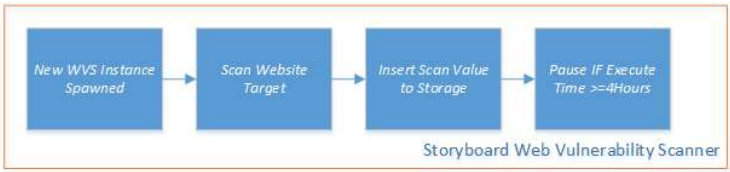
*Melihat dari https://www.websitebuilderexpert.com/how-much-should-a-website-cost/ dalam bagian Maintenance Web

3.3.2 Modul Vulnerability Detection

Modul *Vulnerability Detection* akan melakukan pemindaian terhadap situs web target dan memasukan hasil deteksinya kedalam tempat penyimpanan kemudian ada proses *scheduling* dimana proses ini akan melakukan *check* terhadap jumlah *instance* yang berjalan dan melakukan penjadwalan pemindaian.

Gambar 3.2 Storyboard Scheduler

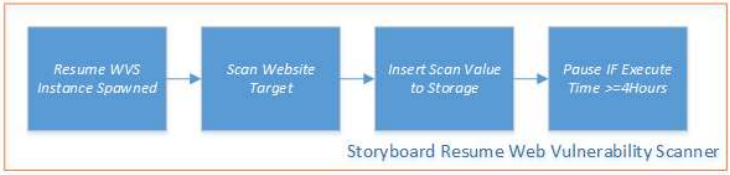
Karena ada batasan dari waktu pemindaian yang hanya dua belas jam maka setelah dua belas jam tadi akan dilakukan pause dari WVS dan hasil pemindaian akan dimasukan kedalam storage yang ada.



Gambar 3.3 Storyboard WVS

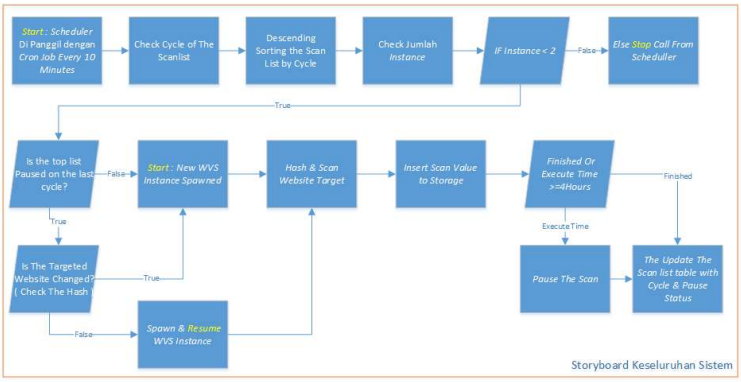
Kemudian apabila semua target website telah berhasil melalui satu '*cycle*' atau satu kali putaran maka *instance* yang statusnya paused tadi akan diresume dengan ketentuan batasan scan **empat jam**.

Gambar 3.4 Storyboard Resume Instance



Gambar 3.5 Storyboard Resume WVS

Dan untuk storyboard keseluruhan system adalah sebagai berikut:



Gambar 3.6 Storyboard Overall

Pada modul ini, ada beberapa file binary yang di keluarkan dari pencarian karena, memiliki kecenderungan *false positive* terhadap *website* yang dipindai. Kategori ekstensi file yang di keluarkan ada pada table berikut:

Table 3.5 Ekstensi yang di exclude

Kategori Ekstensi	Ekstensi
Archive File	zip, rar, tar, 7z
Picture/Image File	img, png, jpg, jpeg
Document File	pdf, doc, docx, rtf, xls, xlsx, ppt, pptx

Halaman ini sengaja dikosongkan)

BAB IV
PERANCANGAN

Pada bab ini dijelaskan perancangan awal yang diperlukan sebelum melakukan penelitian tugas akhir. Bab ini mencakup rancangan yang dibuat dalam persiapan penelitian tugas akhir disertai penjelasannya.

4.1 Kebutuhan Sistem

Kebutuhan sistem didefinisikan dengan melihat kebutuhan dari studi kasus yang digunakan yaitu sistem egovbench baik yang sudah ada maupun yang telah direncanakan dalam perkembangan berikutnya. Dalam tugas akhir ini penulis akan mendefinisikan kebutuhan sistem dalam bidang pemeringkatan website pemda berdasarkan kerentanan yang terdeteksi.

4.1.1 Kebutuhan Spesifik

Kebutuhan spesifik adalah kebutuhan yang didefinisikan berdasarkan kondisi terkini dari sistem yang akan dikembangkan dan permintaan spesifik dari pemegang proyek egovbench.

Table 4.1 Table Kebutuhan Spesifik

Perihal	Stack
Bahasa Pemrograman	PHP7
	Python 3
Website	Laravel
Basis data	Mariadb

Dalam Tabel 4.1 dituliskan perihal bahasa pemrograman yang boleh digunakan adalah PHP7 dan Python 3. PHP7 berkaitan dengan framework Laravel yang digunakan dalam sistem egovbench untuk menampilkan data dan informasi yang digunakan. Python 3 digunakan untuk melakukan *web scrapping*, serta kontrol dan administrasi sistem, namun dalam tugas akhir ini hanya fungsi kontrol dan administrasi sistem saja

yang digunakan. Sedangkan untuk basis data sistem egovbench telah mengunakan mariadb sebagai solusi perangkat lunak yang digunakan.

4.1.2 Kebutuhan Fungsional

Kebutuhan fungsional merupakan kebutuhan yang berhubungan dengan kemampuan yang dimiliki oleh sistem. Sebagai sistem yang bertujuan untuk melakukan pemeringkatan website pemda kebutuhan fungsional ini dibagi menjadi tiga kategori besar yaitu pemindaian, penjadwalan dan pemeringkatan.

4.1.2.1 Kebutuhan Fungsional Pemindaian

Berikut merupakan kebutuhan fungsional dari sub-sistem pemindaian:

- Alat pemindaian yang digunakan adalah framework arachni.
- *Scope* pemindaian berdasarkan pengaturan standar Arachni, dituliskan lebih rinci pada table 4.2.

Table 4.2 Table Scope Pemindaian

No	Scope
1	code injection
2	code injection php input wrapper
3	code injection timing
4	csrf
5	file inclusion
6	ldap injection
7	no sql injection
8	no sql injection differential
9	os cmd injection
10	os cmd injection timing
11	path traversal
12	response splitting
13	rfi
14	session fixation
15	source code disclosure

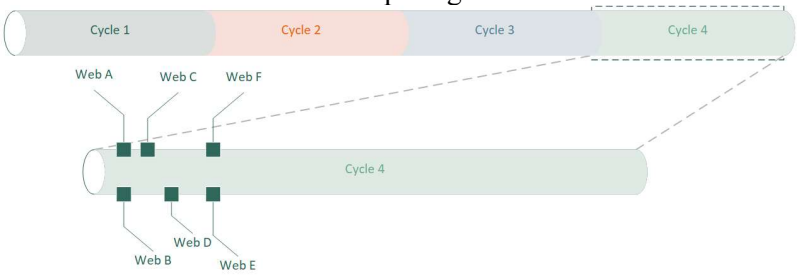
16	sql injection
17	sql injection differential
18	sql injection timing
19	trainer
20	unvalidated redirect
21	unvalidated redirect dom
22	xpath injection
23	xss
24	xss dom
25	xss dom script context
26	xss event
27	xss path
28	xss script context
29	xss tag
30	xxe
31	allowed methods
32	backdoors
33	backup directories
34	backup files
35	captcha
36	common admin interfaces
37	common directories
38	common files
39	cookie set for parent domain
40	credit card
41	cvs svn users
42	directory listing
43	emails
44	form upload
45	hsts
46	htaccess limit
47	html objects
48	http only cookies
49	http put
50	insecure client access policy
51	insecure cookies

52	insecure cors policy
53	insecure cross domain policy access
54	insecure cross domain policy headers
55	interesting responses
56	localstart asp
57	mixed resource
58	origin spoof access restriction bypass
59	password autocomplete
60	private ip
61	ssn
62	unencrypted password forms
63	webdav
64	x frame options
65	xst

4.1.2.2 Kebutuhan Fungsional Penjadwalan

Berikut merupakan kebutuhan fungsional dari sub-sistem pejadwalan:

- Penjadwalan dilakukan secara cycle dalam jangka waktu 3 bulan.
- Pemindaian diberhentikan setelah menyentuh batas waktu yaitu 12 jam.
- Penjadwalan pemindaian dilakukan dengan melihat slot batas maksimum penjadwalan secara simultan. Gambaran secara visual pada gambar 4.1



Gambar 4.1 Gambaran Timeline Target Website dalam Cycle

4.1.2.3 Kebutuhan Fungsional Pemeringkatan

Berikut merupakan kebutuhan fungsional dari sub-sistem pemeringkatan:

- Pemeringkatan berdasarkan hasil pemindaian website pemda.
- Hasil yang diperhitungkan adalah *vulnerability* dengan *severity high*, *medium*, *low* dan *informational*.
- Pemeringkatan dilakukan dengan mengurutkan *severity high*, *medium*, *low* dan *informational* sesuai prioritas urutan dan secara kecil ke besar (*ascending*).

4.1.3 Kebutuhan non Fungsional

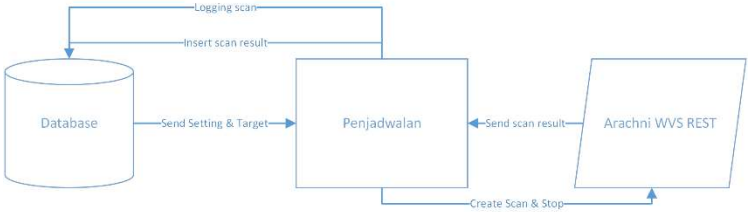
Kebutuhan non Fungsional adalah kebutuhan dari sistem yang tidak berhubungan langsung dengan fungsi dari sistem yang dibangun. Berikut merupakan kebutuhan non fungsional dari sistem pemeringkatan:

- Proses pemindaian jangan sampai memakan semua memory yang tersedia dari sistem *(total system memory 8GB*).

4.2 Desain Sistem

Desain sistem adalah penjelasan bagaimana sistem bekerja.

4.2.1 Desain Penjadwalan



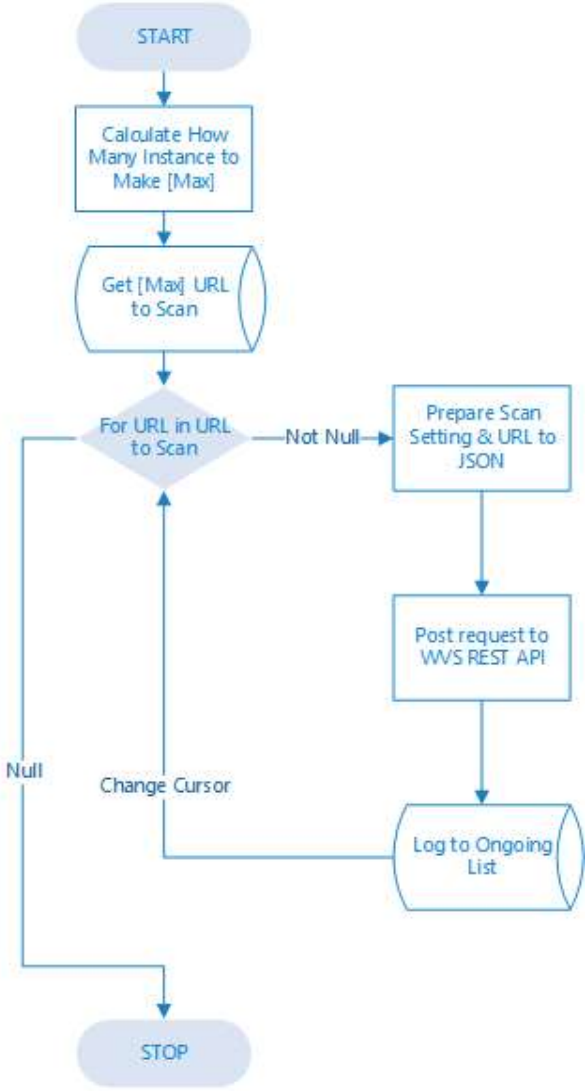
Gambar 4.2 Design Umum Penjadwalan

Desain penjadwalan adalah sekumpulan script/program yang akan melakukan kontrol terhadap alat pemindai (WVS) dan memasukan data kedalam database. Bagian dari kontrol itu adalah membuat perintah pemindaian kepada website pemda, mendapatkan hasil pemindaian yang kemudian dimasukan

kedalam database, dan menghentikan proses pemindaian yang sudah kadaluarsa. Dari keseluruhan bagian control tadi memiliki urutan proses kontrol yang dipanggil sesuai dengan gambar 4.3.

Gambar 4.3 flowchart urutan berjalannya proses

4.2.1.1 Flow Membuat Perintah Pemindaian

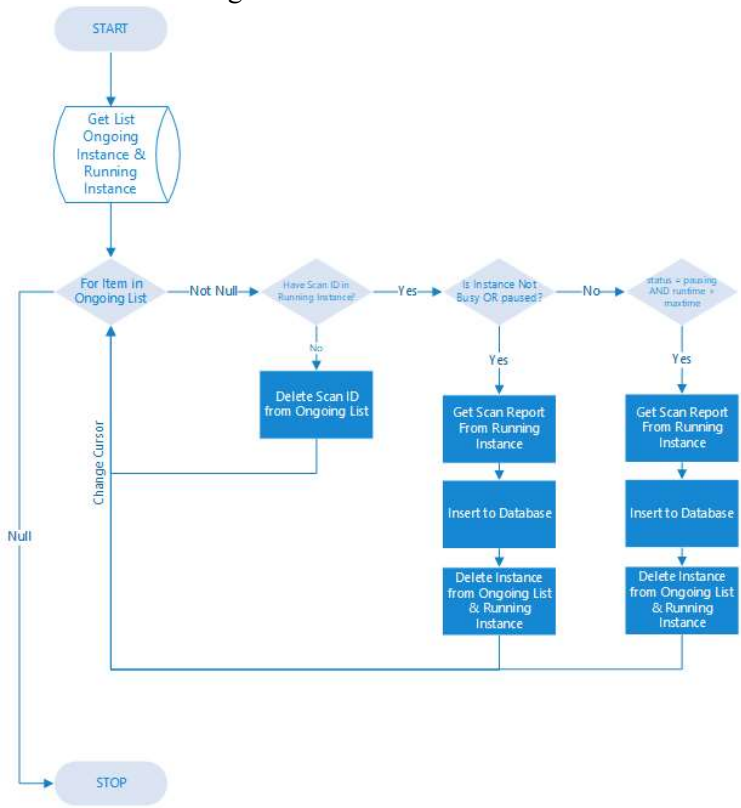


Gambar 4.4 Flowchart Make Scan

Sesuai gambar 4.3 perintah pemindaian dibuat dengan mengambil daftar website dan pengaturan pemindaian dari

database, untuk gunakan oleh alat pemindai (WVS). Perintah pemindaian akan membuat proses pemindaian pada WVS sejumlah maximum proses yang diperbolehkan oleh pengaturan.

4.2.1.2 Flow Pengambilan Hasil Pemindaian

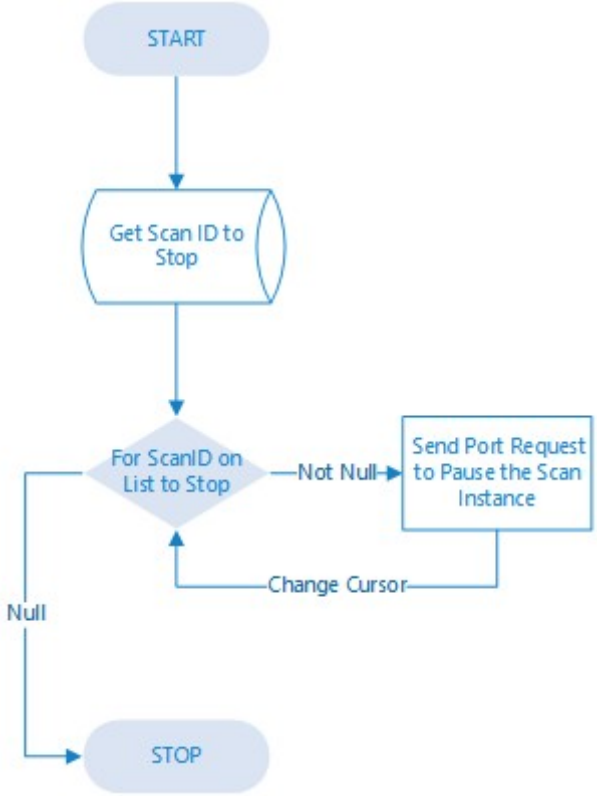


Gambar 4.5 Flowchart Get Scan Result

Sesuai gambar 4.4 pengambilan hasil pemindaian dilakukan dengan melihat daftar pemindaian yang berjalan dari database dan alat pemindaian itu sendiri. Setelah didapatkan daftar pemindaian yang berjalan, proses pengambilan hasil akan melakukan pengecekan apakah proses pemindaian telah selesai melakukan pemindaian dan mendapatkan hasil pemindaian

yang telah selesai. Proses pengambilan hasil ini akan memaksa mengambil hasil pemindaian yang telah melewati batas waktu pemindaian. Setelah mendapatkan hasil pemindaian proses ini akan melakukan pengahapusan proses pemindaian yang masih berada didalam daftar alat pemindai.

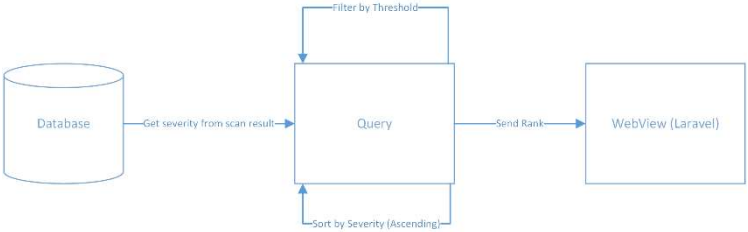
4.2.1.3 Flow Penghentian Proses Pemindaian



Gambar 4.6 Flowchart Stop Scan

Sesuai gambar 4.5 proses penghentian pemindaian berjalan dengan melihat daftar pemindaian yang sudah seharusnya dihentikan, daftar ini dihasilkan dari database dengan melihat waktu pemindaian yang dilakukan.

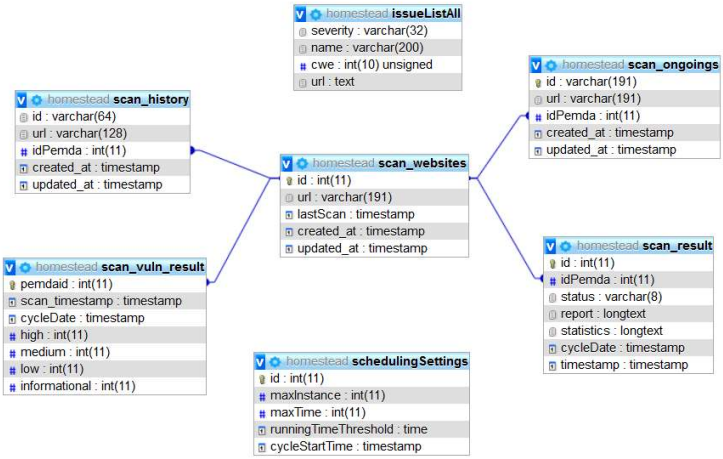
4.2.2 Desain Pemeringkatan



Gambar 4.7 Design Umum Pemeringkatan

Desain pemeringkatan merupakan logika didalam database dimana perintah query akan melakukan pengambilan data dan melakukan pengurutan berdasarkan kriteria yang sudah ditentukan yaitu, berdasarkan prioritas severitynya dari jumlah terkecil ke terbesar. Setelah dilakukan pemeringkatan, query akan melakukan penyaringan berdasarkan batas minimal waktu pemindaian. Adanya batas minimal waktu pemindaian dikarenakan adanya kemungkinan website yang statusnya mati, bisa mendapatkan peringkat baik dibandingkan website yang masih hidup.

4.3 Design Basis Data



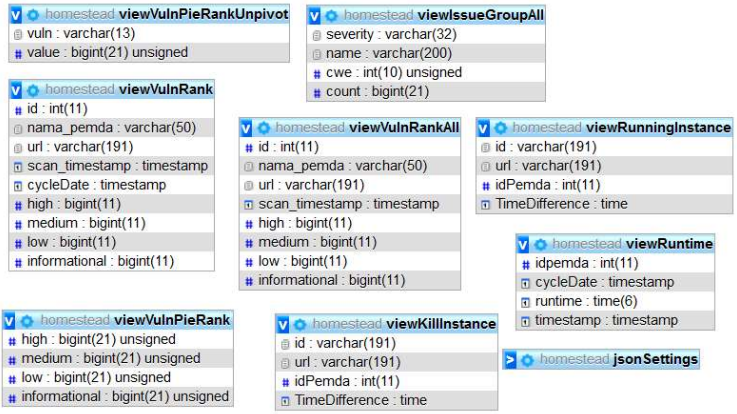
Gambar 4.8 Design Database Table

Design basis data yang akan dibuat akan tampak seperti pada gambar 4.7 secara umum seluruh tabel terhubung dengan key pemda id, dengan table scan_website sebagai pusat dari semua hubungan, secara detail berikut adalah daftar table yang diperlukan:

Table 4.3 Daftar table yang diperlukan

Nama tabel	Kegunaaan
scan_website	Untuk menyimpan daftar website target dan kapan terakhir dilakukan pemindaian.
scan_result	Untuk menyimpan hasil pemindaian oleh WVS.
scan_ongoings	Untuk mencatat proses pemindaian apa yang sedang berlangsung.
scan_history	Untuk mencatat proses pemindaian apa saja yang pernah dilakukan oleh proses penjadwalan.
schedullingSettings	Untuk menyimpan pengaturan dari proses penjadwalan.

jsonSettings	Untuk menyimpan pengaturan yang digunakan oleh WVS dalam melakukan proses pemindaian.
scan_vuln_result	Merupakan *table cache* yang berisi daftar hasil *severity vulnerabilities* yang didapat dari website target dalam bentuk upaya menaikan performa pada *view* di PHP.
issueListAll	Merupakan *table cache* yang berisi daftar hasil keseluruhan *issue* yang ditemukan dari keseluruhan target.



Gambar 4.9 Design Database View + jsonSettings

Selain *table*, penulis juga memerlukan *view* dalam database untuk alasan konsistensi dan kemudahan *development* secara keseluruhan.

Table 4.4 Daftar view yang diperlukan

Nama View	Kegunaaan
viewRunningInstance	Berguna untuk melihat daftar target yang sedang dipindai dan sudah berapa lama dipindai.

viewKillInstance	Berguna untuk melihat daftar pemindaian yang sudah harus dilakukan proses *stop*.
viewVulnRank	Berguna untuk melihat hasil pemeringkatan dengan semua ketentuan termasuk *threshold* waktu *runtime*.
viewVulnRankAll	Berguna untuk melihat hasil pmeringkatan dengan ketentuan yang ada kecuali *threshold runtime.*
viewVulnPieRank	Merupakan *view* yang dibuat berdasarkan *viewVulnRankAll* dengan melakukan SUM semua severity dari target.
viewVulnPieRankUnpivot	Merupakan *view* yang dibuat berdasarkan *viewVulnPieRank* yang dilakukan *unpivot table.*
viewIssueGroupAll	Merupakan *view* dari table *listIssueAll* yang telah di *group by* pada kolom *name.*
viewRuntime	Merupakan view yang dibuat untuk melihat runtime dari hasil pemindaian.

Selain *table* dan *view*, penulis juga memerlukan *function* dalam database untuk mempermudah *development* dalam pengambilan *value* yang sering digunakan untuk fungsi perbandingan.

Table 4.5 Daftar function yang diperlukan

Nama Function	Kegunaaan
getCycleStart	Berguna untuk mendapatkan value dari *cycleStartTime* pada *schedullingSetting*.

getMaxTime	Berguna untuk mendapatkan value dari *maxTime* pada *schedullingSetting*.
getRunningScan	Berguna untuk mendapatkan *running time* dari sebuah hasil pemindaian.
getRunningThreshold	Berguna untuk mendapatkan *value* dari *runningTimeThreshold* pada *schedullingSetting*.

Selain *table, view* dan *function*, penulis juga memerlukan *procedure* dalam database untuk melakukan proses pengolahan data, *filtering* dan *looping procedure* lainnya.

Table 4.6 Daftar procedure yang dibutuhkan

Name Procedure	Kegunaaan
insertScanResult	Untuk memasukan data kedalam *table scan_result*
getTablefromJson	Untuk melihat jumlah *severity* dari sebuah *scan_result*
getSeveritySingleRow	Berfungsi sama dengan *getTablefromJson* namun dilakukan proses input data ke *table scan_vuln_result*
getSeveritySingleTable	Merupakan versi pertama untuk memasukan data dari *getTablefromJson* kedalam table *scan_vuln_result* dari keseluruhan *scan_result (catatan: broken)*
getSeveritySingleTable2	Merupakan versi kedua untuk memasukan data dari *getTablefromJson* kedalam table *scan_vuln_result* dari keseluruhan *scan_result.*

	Procedure ini mengunakan *Looping* terhadap *getSeveritySingleRow.*
getIssueSingleTable	Untuk melihat list dari *Issue* yang ada pada sebuah *scan result.*
setIssue	Berfungsi sama dengan *getIssueSingleTable* namun dilakukan proses input data ke *table listIssueAll*
getIssueAllTable	Berfungsi untuk melakukan Looping terhadap *setIssue* dari seluruh scan result pada cycle ini.

Selain *table, view, function dan procedure*, penulis juga memerlukan *events* dalam database untuk melakukan penjadwalan beberapa *procedure*.

Table 4.7 Daftar events yang dibutuhkan

Name events	Kegunaaan
flush_binary_log	Untuk membersihkan binary log dari MariaDB, mengingat cukup banyak transaksi dalam database ini sehingga membesarkan binary log.
renew_issue_list	Untuk melakukan update pada *table issueListAll* mengunakan *procedure getIssueAlltable.*
renew_vuln_result	Untuk melakukan update pada *table scan_vuln_result* mengunakan *procedure getSeveritySingleTable*

4.4 Use Case

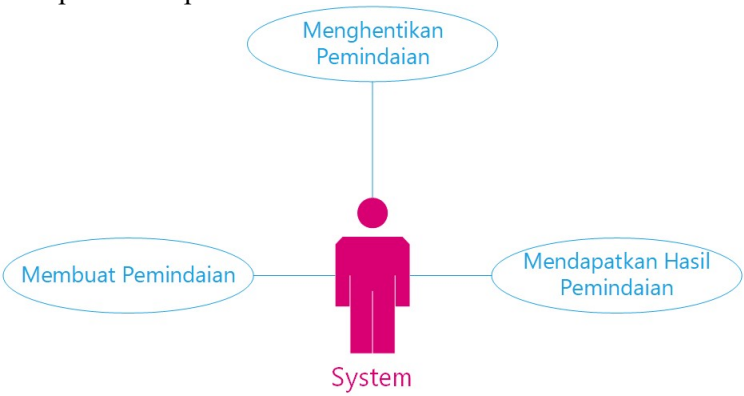
4.4.1 Daftar Use Case

Berikut merupakan daftar *use case* dari modul pemeringkatan kerentanan:

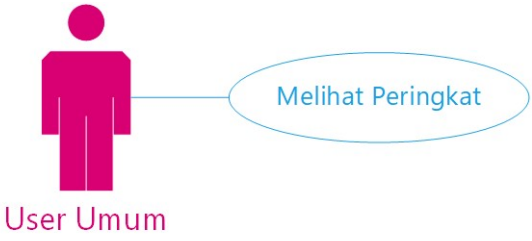
1. Membuat Pemindaian
2. Mendapatkan Hasil Pemindaian
3. Menghentikan Pemindaian
4. Melihat Peringkat
5. Melihat Semua Kerentanan
6. Melihat Kerentanan Pemda
7. Melihat Severity Semua Pemda

4.4.2 Use Case Diagram

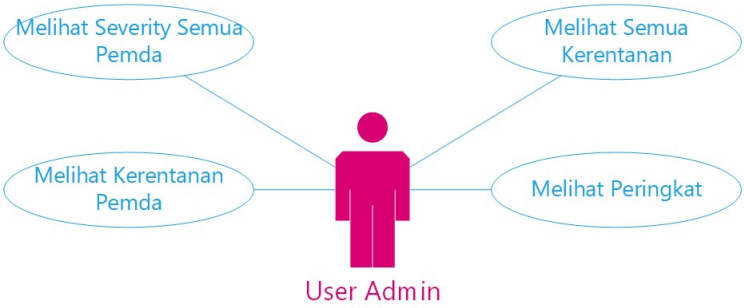
Setelah *use case* teridentifikasi selanjutnya maka dibuat *use case diagram* yang menunjukan hal hal yang dapat dilakukan oleh para aktor pada sistem.



Gambar 4.10 Use Case Diagram System



Gambar 4.11 Use Case Diagram User Umum



Gambar 4.12 Use Case Diagram User Admin

4.4.3 Deskripsi Use Case

Table 4.8 Narasi Use Case Membuat Pemindaian

Use case name	Membuat Pemindaian
Aktor	System
Basic Course	Sistem mendapatkan daftar website yang dipindai dari database, sistem akan membuat perintah pemindaian sejumlah daftar yang didapatkan.
Alternate Course	Jika daftar web pemda dari database kosong maka tidak ada perintah pemindaian yang dibuat.

Table 4.9 Narasi Use Case Mendapatkan Hasil Pemindaian

Use case name	Mendapatkan Hasil Pemindaian
Aktor	System
Basic Course	Sistem mendapatkan daftar proses pemindaian yang ada pada alat pemindaian, sistem akan memeriksa status proses pemindaian apakah sudah dapat diambil, kemudian sistem akan mengambil hasil dari proses yang telah dapat diambil, dan menghapus proses pemindaian dari alat pemindai.
Alternate Course 1	Jika ada proses pemindaian yang belum selesai, namun telah melewati batas waktu ambil paksa hasil pemindaian yang ada dan hapus proses pemindaian pada alat pemindaian.

Alternate Course 2	Jika tidak ada proses pemindaian yang dapat diambil* hasilnya maka *exit*. (*proses pemindaian yang telah selesai atau dihentikan)

Table 4.10 Narasi Use Case Menghetikan Pemindaian

Use case name	Menghentikan Pemindaian
Aktor	System
Basic Course	Sistem akan me
Alternate Course	Jika tidak ada proses yang dapat dihentikan *exit*.

Table 4.11 Narasi Use Case Melihat Peringkat

Use case name	Melihat Peringkat
Aktor	User Umum, User Admin
Basic Course	User memilih menu 'Peringkat' dan memilih kerentanan, sistem akan menampilkan halaman peringkat pemda sesuai kerentanan.
Alternate Course	-

Table 4.12 Narasi Use Case Melihat Severity Pemda

Use case name	Melihat Severity Pemda
Aktor	User Admin
Basic Course	User memilih menu "Data" dan memilih menu "Kerentanan", sistem akan memberikan halaman severity dari pemda.
Alternate Course	User memilih menu "Data Pemda" pada halaman lain pada sub-menu "Data > Kerentanan", sistem menampilkan halaman severity dari pemda.

Table 4.13 Narasi Use Case Melihat Peringkat

Use case name	Melihat Kerentanan Pemda
Aktor	User Admin

Basic Course	User memilih menu “Data” dan memilih menu “Kerentanan”, sistem akan memberikan halaman severity dari pemda, kemudian user memilih “Data Vulnerable” pada kolom “See Detail”
Alternate Course	-

Table 4.14 Narasi Use Case Melihat Peringkat

Use case name	Melihat Semua Kerentanan
Aktor	User Admin
Basic Course	User memilih menu “Data” dan memilih menu “Kerentanan”, sistem akan memberikan halaman severity dari pemda, kemudian user memilih “Data Vulnerable” pada sub-menu “Data > Kerentanan”, system menampilkan halaman vulnerable dari semua pemda.
Alternate Course	User memilih menu “Data Vulnerable” pada halaman lain pada sub-menu “Data > Kerentanan”, sistem menampilkan halaman vulnerable dari semua pemda.

Halaman ini sengaja dikosongkan

BAB V
IMPLEMENTASI

Pada bab implementasi ini dijelaskan hasil implementasi yang telah dilakukan dalam tugas akhir terkait implementasi dari penjadwalan pemindaian, pengaturan *web vulnerabilities scanner*(WVS), dan bentuk database.

5.1 Lingkungan Implementasi

Pada bagian ini dibahas terkait lingkungan pengujian yang digunakan dalam implemetasi tugas akhir terkait perangkat yang digunakan baik perangkat keras maupun perangkat lunak.

5.1.1 Spesifikasi Perangkat Keras

Modul pemeringkatan ini dibuat mengunakan server Lenovo ThinkServer RD350 dengan versi 70QM005NIA untuk spesifikasi lengkap bias dilihat di table 5.1.

Table 5.1 Spesifikasi Perangkat Keras Server

Perihal	**Informasi Perangkat Keras**
Manufacturer	Lenovo
Product Name	ThinkServer RD350
Version	70QM005NIA
Processor	Intel® Xeon® CPU E5-2520 v4
Memory	SK Hynix 8GB RDIMM 2400T-RC1-11
Hard Drive	SAS 10K RPM 300GB

5.1.2 Spesifikasi Perangkat Lunak

Untuk implementasi modul pemeringkatan ini digunakan spesifikasi perangkat lunak sesuai dengan Table 5.2

Table 5.2 Spesifikasi Perangkat Lunak

Tools	
Web Server	Apache 2.4
Server Side	PHP7.2
PHP Framework	Laravel 5.5
Database	MariaDB 10.2
Control Script	Python 3.5

5.2 Penjadwalan Pemindaian

Penjadwalan pemindaian dilakukan dengan memanggil script python untuk melakukan proses makeScan, getScan, dan stopScan. Berikut adalah kode yang digunakan.

```
def getSettings():
    sqll = "SELECT JSON_COMPACT(json) FROM `jsonSettings`"
    dbc.execute(sqll)
    dbresult = dbc.fetchone()
    jsonSettings = dbresult[0].decode("utf-8")
    return jsonSettings

def getScanUrl(limit):
    sqll = "SELECT id, url FROM scan_websites JOIN( SELECT idPemda,
MAX(updated_at) as ScanMaxTime FROM `scan_history` GROUP BY idPemda )
sub ON scan_websites.id = sub.idPemda WHERE scan_websites.lastScan <
getCycleStart() AND sub.ScanMaxTime <= NOW() - INTERVAL 1 DAY ORDER
BY RAND() LIMIT {}".format(limit)
    dbc.execute(sqll)
    dbresult = dbc.fetchall()
    return dbresult

def getMaxInstance():
    sqll = "SELECT `maxInstance` FROM `schedulingSettings`"
    dbc.execute(sqll)
    dbresult = dbc.fetchone()
    return dbresult[0]

def logScan(url,idscan,idpemda):
    sqll = "INSERT INTO `scan_ongoings` (`id`,`url`,`idPemda`) VALUES
('{}', '{}','{}');".format(idscan,url,idpemda)
    dbc.execute(sqll)
    sqll = "INSERT INTO `scan_history` (`id`,`url`,`idPemda`) VALUES
('{}', '{}','{}');".format(idscan,url,idpemda)
    dbc.execute(sqll)
    db.commit()

def getScans():
    r = requests.get(scansURL, auth=auth)
    return r.text

def postScans(jsonSettings):
    r = requests.post(scansURL, auth=auth, json=jsonSettings)
    return r.text
```

Kode 5.1 makeScan.py

Berdasarkan kode 5.1 MakeScan berguna untuk membuat pemindaian dengan mengambil daftar website dan pengaturan pemindaian dari database untuk gunakan oleh WVS. *Script* ini juga akan membuat proses pemindaian sesuai dengan

MaxInstance yang telah ditentukan. Dalam table 5.3 terdapat daftar fungsi yang ada pada makeScan:

Table 5.3 Daftar Fungsi makeScan

Fungsi	Penjelasan
getSettings	Fungsi ini berguna untuk mengambil setting pemindaian dari database.
getScanUrl	Berguna untuk mendapatkan website target.
getMaxInstance	Berguna untuk mendapatkan jumlah maksimum pemindaian yang boleh dilakukan.
logScan	Berguna untuk logging kedalam database
getScans	Berguna untuk mendapatkan daftar proses pemindaian dari alat pemindaian (WVS)
postScans	Berguna untuk mengirim perintah pemindaian ke alat pemindaian (WVS)

```
def logResult(idscan,idpemda,status,report,statistics):
    # escape the text & JSON
    es_status = mariadb.escape_string(status).decode()
    es_report = mariadb.escape_string(report).decode()
    es_statis = mariadb.escape_string(statistics).decode()
    sqll = "CALL
`insertScanResult`('{}','{}','{}','{}');".format(idpemda,es_status,es
_report,es_statis)
    # print(sqll)
    dbc.execute(sqll)
    db.commit()

def getIDfromDB():
    sqll = "SELECT id,idpemda FROM `scan_ongoings`"
    dbc.execute(sqll)
    dbresult = dbc.fetchall()
    return dbresult

def deleteScans(idscan):
    url = '{}/{}'.format(scansURL, idscan)
    r = requests.delete(url, auth=auth)
    sqll = "DELETE FROM `scan_ongoings` WHERE `scan_ongoings`.`id` =
'{}';".format(idscan)
    dbc.execute(sqll)
```

```
    db.commit()
    return r.text

def getScans():
    r = requests.get(scansURL, auth=auth)
    return r.text

def getSummary(idscan):
    url = '{}/{}/summary'.format(scansURL, idscan)
    r = requests.get(url, auth=auth)
    return r.text

def getReport(idscan):
    url = '{}/{}/report'.format(scansURL, idscan)
    r = requests.get(url, auth=auth)
    return r.text

def deleteIDfromDB(idscan):
    url = '{}/{}'.format(scansURL, idscan)
    sqll = "DELETE FROM `scan_ongoings` WHERE `scan_ongoings`.`id` =
'{}';".format(idscan)
    dbc.execute(sqll)
    db.commit()

def getMaxTime():
    sqll = "SELECT maxTime FROM `schedulingSettings`"
    dbc.execute(sqll)
    db.commit()
    dbresult = dbc.fetchone()
    return dbresult
```

Kode 5.2 getScan.py

Dapat dilihat pada Kode 5.2 *script* GetScan berguna untuk mengambil hasil pemindaian dari proses yang telah selesai atau dihentikan oleh stopScan kemudian proses pemindaian akan dihapus dari database dan alat pemindaian. Dalam table 5.4 terdapat daftar fungsi yang ada pada makeScan:

Table 5.4 Daftar Fungsi getScan

Fungsi	Penjelasan
logResult	Beguna untuk memasukan hasil pemindaian kedalam database.
getIDfromDB	Berguna untuk mendapatkan ID dari proses pemindaian.
deleteScans	Berguna untuk mendapatkan jumlah maksimum pemindaian yang boleh dilakukan.

getSummary	Beguna untuk mendapatkan status pemindaian dari alat pemindaian.
getScans	Berguna untuk mendapatkan daftar proses pemindaian dari alat pemindaian (WVS).
getReport	Berguna untuk mendapatkan hasil pemindaian dari proses pemindaian.
deleteIDfromDB	Berguna untuk menghapus ID dari proses pemindaian yang berada dalam daftar proses yang sedang berjalan di database.
getMaxTime	Berguna untuk mendapatkan waktu maksimum sebuah proses pemindaian dapat berjalan.

```
def getScans():
    r = requests.get(scansURL, auth=auth)
    return r.text

def pauseScans(idscan):
    url = '{}/{}/pause'.format(scansURL, idscan)
    r = requests.put(url, auth=auth)
    return r.text

def getSummary(idscan):
    url = '{}/{}/summary'.format(scansURL, idscan)
    r = requests.get(url, auth=auth)
    return r.text

def getID():
    sqll = "SELECT id,url FROM `viewKillInstance`"
    dbc.execute(sqll)
    dbresult = dbc.fetchall()
    return dbresult
```

Kode 5.3 stopScan.py

Dapat dilihat pada Kode 5.3 StopScan berguna untuk menghentikan proses pemindaian dari daftar proses yang harus dihentikan yang berada di database. Dalam table 5.5 terdapat daftar fungsi yang ada pada makeScan:

Table 5.5 Daftar Fungsi stopScan

Fungsi	Penjelasan
getScans	Berguna untuk mendapatkan daftar proses pemindaian dari alat pemindaian (WVS).
pauseScans	Berguna untuk memberhentikan proses pemindaian.
getSummary	Berguna untuk mendapatkan status proses pemindaian dari alat pemindaian (WVS)
getID	Berguna untuk mendapatkan daftar proses yang harus diberhentikan pada alat pemindaian.

```
protected function schedule(Schedule $schedule)
    {
        // $schedule->command('inspire')
        //          ->hourly();
        $schedule->exec('python3 <redacted>/egovuln.py/stopScan.py')-
>everyFiveMinutes()->appendOutputTo('<redacted>/sched.log/stop');
        $schedule->exec('python3 <redacted>/egovuln.py/getScan.py')-
>everyFiveMinutes()->appendOutputTo('<redacted>/sched.log/get');
        $schedule->exec('python3 <redacted>/egovuln.py/makeScan.py')-
>everyFiveMinutes()->appendOutputTo('<redacted>/sched.log/make');
    }
```

Kode 5.4 cronjob Laravel kernel

Dapat dilihat pada Kode 5.4 Laravel Kernel berguna untuk memangil kode python setiap waktu yang ditentukan sesuai keinginan dengan bantuan cronjob.

BAB VI
HASIL DAN PEMBAHASAN

Bab ini menjelaskan mengenai hasil analisa dari implementasi penelitian yang telah dilaksanakan.

6.1 Hasil Pengujian

Pada tahap ini menjelaskan tentang pengujian modul pada aplikasi yang dibuat. Pengujian modul yang dilakukan mengunakan test case sebagai alat pengujian. Test case berisi pengujian terhadap logic aplikasi dan hasil yang ditampikan.

6.1.1 Schedulling Test

Tes ini melihat dari perilaku scheduler yang dibuat apakah sudah sesuai dengan yang diinginkan dengan melihat dari hasil yang tercatat pada table scan_result.

idPemda	cycleDate	timestamp
81	2017-12-29 13:11:07	2017-12-29 13:11:07
90	2017-12-29 13:11:07	2017-12-30 20:31:05
292	2017-12-29 13:11:07	2018-01-03 14:33:09
242	2017-12-29 13:11:07	2018-01-03 14:33:30

Gambar 6.1 test pertama scheduling

Dapat dilihat pada gambar 6.1, semua sesuai dengan yang diinginkan karena tidak ada idPemda yang terlewat dan tidak ada data yang ganda. (catatan: daftar pemda yang setuju saat itu berjumlah empat saja)

idPemda	cycleDate	timestamp
259	2018-06-14 16:15:01	2018-06-14 16:15:01
81	2018-06-14 16:15:01	2018-06-14 16:15:02
90	2018-06-14 16:15:01	2018-06-14 16:15:03
242	2018-06-14 16:15:01	2018-06-14 16:20:01
240	2018-06-14 16:15:01	2018-06-14 16:20:02
271	2018-06-14 16:15:01	2018-06-14 16:20:02
63	2018-06-14 16:15:01	2018-06-14 16:25:02
178	2018-06-14 16:15:01	2018-06-15 04:00:05
13	2018-06-14 16:15:01	2018-06-15 04:55:07
13	2018-06-14 16:15:01	2018-06-15 04:55:12
292	2018-06-14 16:15:01	2018-06-15 05:05:17
274	2018-06-14 16:15:01	2018-06-15 06:00:09
274	2018-06-14 16:15:01	2018-06-15 07:15:09
367	2018-06-14 16:15:01	2018-06-15 17:05:54
367	2018-06-14 16:15:01	2018-06-15 18:15:48
367	2018-06-14 16:15:01	2018-06-15 19:40:47

Gambar 6.2 test kedua schedulling

Seperti dilihat pada gambar 6.2, pada test kedua ini didapati ada *bug* pada *scheduler* karena ada data ganda ya itu pada pemdaid 13,274, dan 367. Hal ini disebabkan logic yang digunakan pada test pertama adalah scanning dilakukan satu satu tidak secara parallel. (catatan:daftar pemda yang telah menyetujui untuk kerjasama dalam hal pengetesan ini meningkat menjadi dua belas)

idPemda	cycleDate	timestamp
63	2018-10-31 07:58:06	2018-10-31 08:11:20
271	2018-10-31 07:58:06	2018-10-31 08:20:02
242	2018-10-31 07:58:06	2018-10-31 08:25:02
274	2018-10-31 07:58:06	2018-10-31 09:25:10
240	2018-10-31 07:58:06	2018-10-31 09:30:02
90	2018-10-31 07:58:06	2018-10-31 09:35:01
13	2018-10-31 07:58:06	2018-10-31 20:15:08
178	2018-10-31 07:58:06	2018-10-31 20:36:14
292	2018-10-31 07:58:06	2018-10-31 20:40:02
259	2018-10-31 07:58:06	2018-10-31 20:45:02
367	2018-10-31 07:58:06	2018-10-31 21:47:33
81	2018-10-31 07:58:06	2018-11-01 08:26:22

Gambar 6.3 test ketiga scheduling

Seperti dapat dilihat pada gambar 6.3 bug dapat diatasi dengan penambahan logic baru berupa *double check* mengunakan "scan_websites.lastScan" dan "scan_history.created_at" pada query yang dilakukan oleh scheduler ke database.

```
24   def getScanUrl(limit):
25 -     sqll = "SELECT id, url FROM scan_websites WHERE
     scan_websites.lastScan < getCycleStart() ORDER BY RAND() LIMIT
     {}".format(limit)
26       dbc.execute(sqll)
27       dbresult = dbc.fetchall()
28       return dbresult
24   def getScanUrl(limit):
25 +     sqll = "SELECT id, url FROM scan_websites JOIN( SELECT
     idPemda, MAX(updated_at) as ScanMaxTime FROM `scan_history` GROUP
     BY idPemda ) sub ON scan_websites.id = sub.idPemda WHERE
     scan_websites.lastScan < getCycleStart() AND sub.ScanMaxTime <=
     NOW() - INTERVAL 1 DAY ORDER BY RAND() LIMIT {}".format(limit)
26       dbc.execute(sqll)
27       dbresult = dbc.fetchall()
28       return dbresult
```

Gambar 6.4 histori perubahan untuk perbaikan bug

6.1.2 Accuracy Test

Tes ini melihat dari keseluruhan view dan table akurat sesuai dengan perhitungan, pemeringkatan dan fakta dilapangan.

- Options

id	nama_pemda	high	medium	low	informational
13	KAB. ACEH BARAT DAYA	0	0	1	12
274	KOTA MADIUN	0	1	4	6
178	KAB. CIREBON	0	83	2	4
367	KAB. HULU SUNGAI UTARA	1	84	5	2
81	KAB. INDRAGIRI HULU	3	3	2	26

Gambar 6.5 viewVulnRank

Dapat dilihat pada gambar 6.5, hanya terdapat 5 pemda yang berhasil masuk kedalam list, dikarenakan adanya threshold lama pemindaian yaitu satu jam, jadi apabila hasil pemindaian tidak lebih lama dari satu jam akan dihilangkan dari view.

+ Options

id	nama_pemda	high	medium	low	informational
271	KOTA PROBOLINGGO	0	0	0	0
63	KAB. TANAH DATAR	0	0	0	0
259	KAB. NGAWI	0	0	1	1
90	KOTA PEKAN BARU	0	0	1	3
242	KAB. TULUNGAGUNG	0	0	1	8
13	KAB. ACEH BARAT DAYA	0	0	1	12
240	KAB. PONOROGO	0	1	1	0
292	KAB. BANGLI	0	1	1	2
274	KOTA MADIUN	0	1	4	6
178	KAB. CIREBON	0	83	2	4
367	KAB. HULU SUNGAI UTARA	1	84	5	2
81	KAB. INDRAGIRI HULU	3	3	2	26

Gambar 6.6 viewVulnRankAll

Dapat dilihat pada gambar 6.6, semua pemda yang berhasil masuk kedalam list, apabila tidak adanya threshold lama pemindaian.

severity	name	cwe	count
high	Cross-Site Request Forgery	352	4
medium	Common directory	538	165
medium	Missing 'Strict-Transport-Security' header	200	2
medium	Backup file	530	5
medium	Unencrypted password form	319	1
low	Missing 'X-Frame-Options' header	693	10
low	Common sensitive file	*NULL*	2
low	Common administration interface	*NULL*	7
informational	Interesting response	*NULL*	54
informational	Allowed HTTP methods	*NULL*	1
informational	HTML object	200	6
informational	HttpOnly cookie	200	2
informational	Cookie set for parent domain	200	1

Gambar 6.7 viewIssueGroupAll

Dapat dilihat pada gambar 6.7, semua issue yang ada pada semua target terlihat dan dapat terhitung jumlah dari masing masing issue yang ada.

high	medium	low	informational
4	173	19	64

Gambar 6.8 viewVulnPieRank

Dapat dilihat pada gambar 6.7, merupakan perhitungan keseluruhan severity dari masing masing kategori.

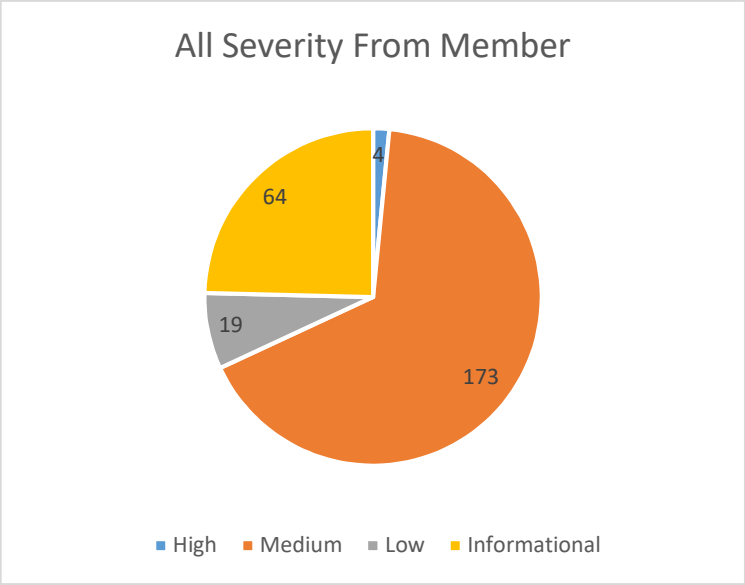
Dari beberapa view penting yang sudah dites pada test case ini tidak terdapat bug logic maupun hasil, dengan cara check manual pada hasil scan yang terdapat pada “scan_result.report”.

6.2 Pembahasan

Pada tahap ini menjelaskan tentang temuan temuan yang ada pada hasil pengerjaan tugas akhir ini.

6.2.1 Pembahasan severity

Seperti dapat dilihat pada gambar 6.7 pada sub-bab sebelumnya dapat dilihat jumlah terbanyak terdapat pada *severity* medium dengan jumlah keseluruhan 173 diikuti oleh *severity informational* dengan jumlah keseluruhan 64 diikuti severity low dengan 19 dan severity high dengan 4.



Gambar 6.9 Pie Chart semua severity

Dan untuk lebih jelas lagi dapat dilihat pie chart pada gambar 6.8 severity medium memiliki persentasi hingga lebih dari 60% dari total. Sangat disayangkan masih terdapat severity high pada keseluruhan target.

6.2.2 Pembahasan vulnerability

severity	name	cwe	count
high	Cross-Site Request Forgery	352	4
medium	Common directory	538	165
medium	Missing 'Strict-Transport-Security' header	200	2
medium	Backup file	530	5
medium	Unencrypted password form	319	1
low	Missing 'X-Frame-Options' header	693	10
low	Common sensitive file	NULL	2
low	Common administration interface	NULL	7
informational	Interesting response	NULL	54
informational	Allowed HTTP methods	NULL	1
informational	HTML object	200	6
informational	HttpOnly cookie	200	2
informational	Cookie set for parent domain	200	1

Gambar 6.10 List dari keseluruhan vulnerability yang didapatkan

Dapat dilihat pada gambar 6.10 diatas, *severity medium* sangat banyak didapatkan dari *vulnerability* 'common directory'. Dan untuk bagian *severity informational* sangat banyak didapatkan dari *vulnerability* 'interesting response'. Sedangkan pada severity high didapatkan dari vulnerability 'CSRF'.

6.2.3 Pembahasan pemeringkatan

Pemeringkatan dilakukan dengan melihat tingkat severity yang terdeteksi dari hasil pindai WVS dan kemudian dilakukan filtering terhadap hasil sehingga hanya website dengan runtime pemindaian selama satu jam saja yang dapat terlihat. Hal ini dilakukan karena website yang memiliki runtime kurang dari memiliki hasil yang sangat jomplang dari yang memiliki runtime satu jam lebih yang dapat dilihat pada gambar 6.11.

idpemda	high	medium	low	informational	runtime
13	0	0	1	12	12:14:53
178	0	83	2	4	12:09:58
367	1	84	5	2	12:09:57
81	3	3	2	26	12:09:49
274	0	1	4	6	01:24:56
63	0	0	0	0	00:11:16.
271	0	0	0	0	00:04:58
242	0	0	1	8	00:04:57
259	0	0	1	1	00:04:57
90	0	0	1	3	00:04:56
240	0	1	1	0	00:04:49
292	0	1	1	2	00:03:43

Gambar 6.11 pemeringkatan semua dengan runtime

#	Nama Pemda	High	Medium	Low	Informational
1	KAB. ACEH BARAT DAYA	0	0	1	12
2	KOTA MADIUN	0	1	4	6
3	KAB. CIREBON	0	83	2	4
4	KAB. HULU SUNGAI UTARA	1	84	5	2
5	KAB. INDRAGIRI HULU	3	3	2	26

Gambar 6.12 pemeringkatan pemda dengan threshold

Dapat dilihat dari gambar 6.12, Kabupaten Aceh Barat Daya memperoleh peringkat pertama dikarenakan website pemda tersebut tidak memiliki *severity medium dan high* sedangkan pemda lain memiliki *severity medium.*

BAB VII
KESIMPULAN DAN SARAN

Pada bab ini menjelaskan tentang kesimpulan dan saran dari hasil pengerjaan tugas akhir ini. kesimpulan dan saran diharapkan berguna untuk pengembangan selanjutnya.

7.1 Kesimpulan

Berdasarkan hasil penelitian tugas akhir ini, maka dapat disimpulkan sebagai berikut :

1. Modul Penjadwalan(scheduling) berjalan dengan baik sesuai dengan keinginan. Terbukti dari scheduling test ketiga.
2. Modul Pemeringkatan berhasil dibuat dan berjalan dengan baik sesuai keinginan. Terbukti dari hasil accuracy test pada semua view yang berkaitan dengan pemeringkatan.
3. Website pemerintah daerah yang telah dipindai hanya satu website saja yang memiliki *severity high* dari dua belas website yang mau bekerjasama dengan egovbench. Hal ini sudah baik namun masih bias ditingkatkan dengan menekan *severity high* menjadi 0 dan menekan *severity medium* menjadi lebih kecil lagi.
4. Modul Rekomendasi Dana tidak dapat diimplementasikan dikarenakan keterbatasan waktu, metode dan informasi yang ada. Dari keseluruhan website yang bersedia dipindai memiliki keragaman yang sangat mencolok, dimana ada website yang bentuknya masih sebagai tempat berita dan informasi publik hingga website yang sudah menjadi portal benyak Sistem Informasi Masyarakat.

7.2 Saran

Beberapa saran yang dapat dipertimbangkan untuk penelitian lebih lanjut adalah sebagai berikut :

1. Modul rekomendasi dana menurut penulis memerlukan penilaian menggunakan Common Vulnerability Scoring System (CVSS), dimana hanya orang dalam organisasi tersebut yang dapat menilai hal tersebut, faktor lain adalah biaya perbaikan di setiap daerah berbeda-beda sehingga menyulitkan rekomendasi secara menyeluruh. Melihat kompleksitas feature ini dapat menjadi judul tugas akhir untuk mahasiswa lain.
2. Apabila website pemda yang menjadi target sudah mencapai lebih dari seratus *script* penjadwalan dapat dipercepat performanya mengunakan *async_task.*
3. Perlu adanya sistem/program baru untuk mencari tahu apakah *home directory website* target berubah secara *routing* ataupun *dns*, contohnya:
    a. link lama: www.ponorogo.gov.id menjadi
    b. link baru: portal.ponorogokab.gov.id atau www.ponorogokab.gov.id/home

Daftar Pustaka

[1] "Monitoring Web Pemda App," ADDI SI ITS, 9 Mei 2017. [Online]. Available: http://egovbench.addi.is.its.ac.id/methodology3.php. [Accessed 1 Oktober 2017].

[2] "Vulnerability (Computing)," Wikipedia, 17 September 2017. [Online]. Available: https://en.wikipedia.org/wiki/Vulnerability_(computing). [Accessed 20 September 2017].

[3] "Information assurance," Wikipedia, 14 Agustus 2017. [Online]. Available: https://en.wikipedia.org/wiki/Information_assurance. [Accessed 20 September 2017].

[4] "HELP AND FAQ - W3C," W3C, [Online]. Available: https://www.w3.org/Help/#webinternet. [Accessed 20 September 2017].

[5] "SQL Injection," Microsoft, [Online]. Available: https://technet.microsoft.com/en-us/library/ms161953%28v=SQL.105%29.aspx. [Accessed 20 September 2017].

[6] "SQL Injection," Wikipedia, 17 September 2017. [Online]. Available: https://en.wikipedia.org/wiki/SQL_injection. [Accessed 20 September 2017].

[7] "Cross-site Scripting (XSS)," OWASP, 4 Juni 2016. [Online]. Available: https://www.owasp.org/index.php/Cross-site_Scripting_(XSS). [Accessed 20 September 2017].

[8] "Cross-Site Request Forgery," OWASP, 20 Juni 2017. [Online]. Available: https://www.owasp.org/index.php/Cross-Site_Request_Forgery_(CSRF). [Accessed 20 September 2017].

[9] "File Inclusion Vulnerability," Wikipedia, 7 Agustus 2017. [Online]. Available:

https://en.wikipedia.org/wiki/File_inclusion_vulnerability. [Accessed 20 September 2017].

[10] "Unvalidated Redirects and Forwards Cheat Sheet," OWASP, 11 September 2017. [Online]. Available: https://www.owasp.org/index.php/Unvalidated_Redirects_and_Forwards_Cheat_Sheet. [Accessed 20 September 2017].

[11] "Review Old, Backup and Unreferenced Files for Sensitive Information," OWASP, [Online]. Available: https://www.owasp.org/index.php/Review_Old,_Backup_and_Unreferenced_Files_for_Sensitive_Information_(OTG-CONFIG-004). [Accessed 20 September 2017].

[12] "Backup files," Acunetix, [Online]. Available: https://www.acunetix.com/vulnerabilities/web/backup-files. [Accessed 20 September 2017].

[13] "Path Traversal," OWASP, 6 Oktober 2015. [Online]. Available: https://www.owasp.org/index.php/Path_Traversal. [Accessed 20 September 2017].

[14] "Command Injection," OWASP, 7 September 2016. [Online]. Available: https://www.owasp.org/index.php/Command_Injection. [Accessed 20 September 2017].

[15] "Code Injection," OWASP, 31 December 2013. [Online]. Available: https://www.owasp.org/index.php/Code_Injection. [Accessed 20 September 2017].

[16] "Incompatibilities and Feature Differences Between MariaDB 10.2 and MySQL 5.7," MariaDB, February 2017. [Online]. Available: https://mariadb.com/kb/en/library/incompatibilities-and-feature-differences-between-mariadb-102-and-mysql-57/. [Accessed November 2017].

BIODATA PENULIS

Penulis lahir di Banjarmasin pada tanggal 10 Desember 1995. Penulis merupakan anak pertama dari dua bersaudara. Penulis telah menempuh pendidikan formal di sekolah negeri mulai dari SDN Teluk Dalam 3 Banjarmasin hingga lulus pada tahun 2007, SMPN 1 Banjarmasin hingga lulus pada tahun 2010, dan SMAN 1 Banjarmasin hingga lulus pada tahun 2013. Setelah lulus, penulis melanjutkan ke jenjang perguruan tinggi negeri di Surabaya, yakni Departemen Sistem Informasi Institut Teknologi Sepuluh Nopember Surabaya. Sebagai mahasiswa penulis aktif dalam urusan akademik, non akademik. Tercatat penulis pernah menjadi staff dan staff ahli pada Departemen *Information Media* di Badan Eksekutif Mahasiswa Fakultas Teknologi Informasi (BEM FTIf) ITS Surabaya. Selain organisasi formal, penulis juga pernah mengikuti organisasi non-formal, yakni menjadi Angota & Pengajar pada *Information System Geeks Community / Network Security* Departemen Sistem Informasi. Selain organisasi, penulis juga aktif dalam kepanitiaan, baik panitia dalam organisasi yang diikutinya, maupun di luar organisasi. Penulis juga pernah menjalani kerja praktik di PT Bank Rakyat Indonesia, Tbk di Surabaya selama 2 bulan pada tahun 2016.
Untuk mendapatkan gelar Sarjana Komputer (S.Kom), penulis mengambil laboratorium bidang minat Infrastruktur dan Keamanan Teknologi Informasi (IKTI). Untuk kepentingan penelitian penulis juga dapat dihubungi melalui e-mail: mchmmdrizki@gmail.com.